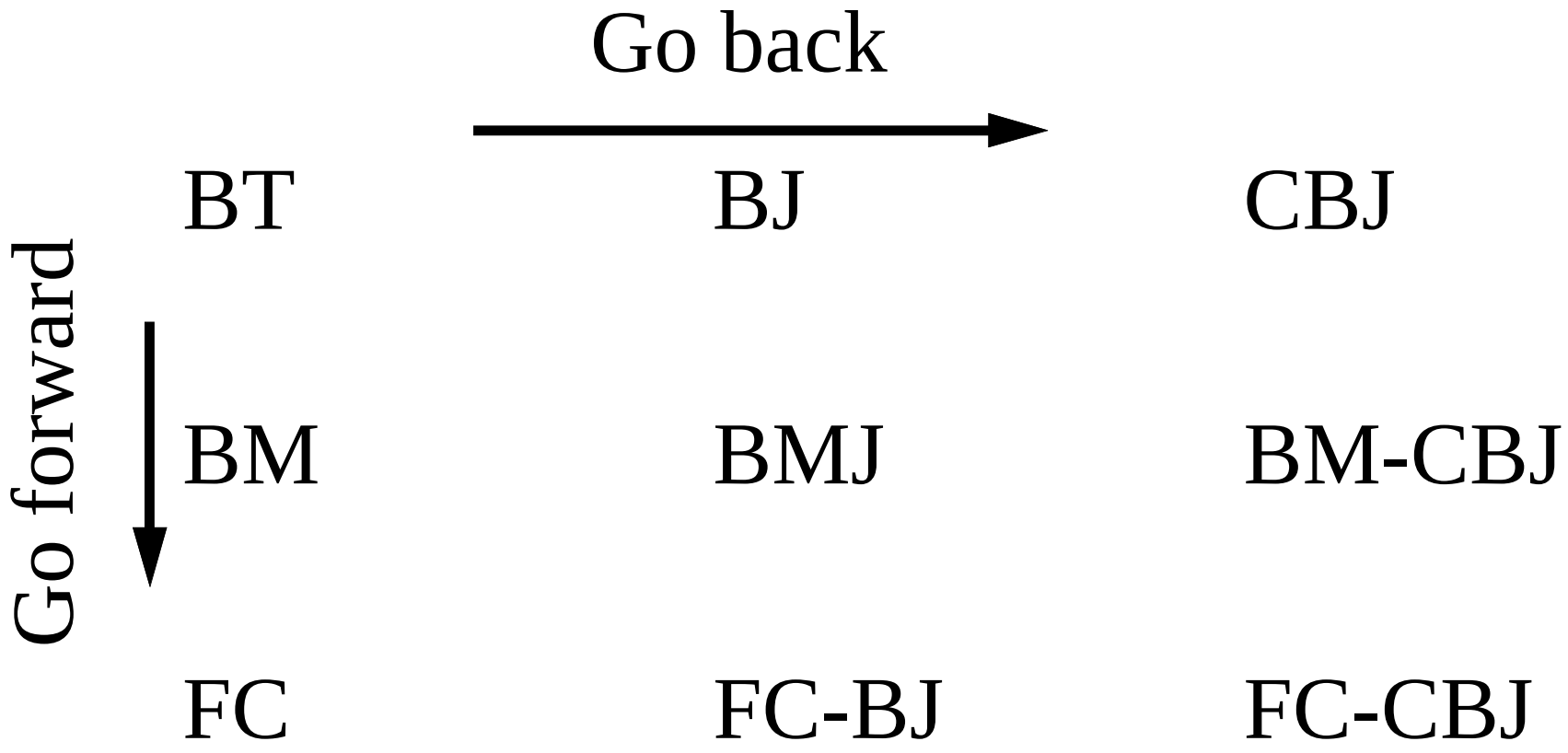
# Backtrack search algorithms



# Backtrack search

**procedure** *bcssp*(*n*)

*consistent = true*

*i = initialize*()

**loop**

**if** *consistent* **then** (*i*, *consistent*) = *label*(*i*)

**else** (*i*, *consistent*) = *unlabel*(*i*)

**if** $i > n$ **then return** "solution found"

**else if** $i = 0$ **then return** "no solution"

**endloop**

**end** *bcssp*

# Chronological backtracking: *label*

**function** *bt-label*(*i*)
    **for** each $v_k \in CD_i$ **do**
        Set $x_i = v_k$ and *consistent = true*
        **for** *j* **from** *1* **to** *i–1* **do** /* Previously assigned variables*/
            **if** $\neg C_{ij}(x_i, x_j)$ **then**
                Remove $v_k$ from $CD_i$ and set *consistent = false*
                Unassign $x_i$ and **break** inner loop
            **endif**
        **if** *consistent* **then** **return** (*i+1*, *true*)
    **endfor**
    **return** (*i*, *false*)
**end** *bt-label*

# Chronological backtracking: *unlabel*

**function** *bt-unlabel*(*i*)

*h = i – 1* /* Backtrack to previous variable */

$CD_i = D_i$

Remove current value assigned to $x_h$ from $CD_h$

Unassign $x_h$

**if** $CD_h$ is empty **then**

**return** (*h*, *false*)

**else**

**return** (*h*, *true*)

**end** *bt-unlabel*

# Backtrack search in N-queens

- Figure 2 from Kondrak & van Beek 1997

# Backjumping

- *max-check*$_i$ (initialized to *0*)
  - deepest variable with which variable $x_i$ performed a consistency check
  - if $x_i$ has no consistent values, then jump back to *max-check*$_i$

# Backjumping: *label*

## Modification to *bt-label*

**function** *bj-label*(*i*)

**for** each $v_k \in CD_i$ **do**

Set $x_i = v_k$ and *consistent = true*

**for** *j* **from** *1* **to** *i–1* **do** /* for each past variable */

$max\text{-}check_i = max(max\text{-}check_i, j)$

**if** $\neg C_{ij}(x_i, x_j)$ **then**

Remove $v_k$ from $CD_i$ and set *consistent = false*

Unassign $x_i$ and **break** inner loop

**endif**

**endfor**

**if** *consistent* **then** **return** (*i+1*, *true*)

**endfor**

**return** (*i*, *false*)

**end** *bj-label*

# Backjumping: *unlabel*

Modification of *bt-unlabel*

**function** *bj-unlabel*(*i*)

```
    h = max-check_i
    for j from h+1 to i do
        max-check_j = 0
        CD_j = D_j
    endfor
    Remove current value assigned to x_h from CD_h
    Unassign x_h
    if CD_h is empty then      return (h, false)
    else                       return (h, true)
end bj-unlabel
```

# Conflict-directed backjumping

- *conf-set*$_i$ (initialized to $\{0\}$)
  - set of past variables with which $x_i$ conflicts
  - $h$ is in *conf-set*$_i$ if variable $x_h$ played a role in ruling out a value for $x_i$

# Conflict-directed backjumping: *label*

## Modification to *bj-label*

**function** *cbj-label*(*i*)

**for** each $v_k \in CD_i$ **do**

Set $x_i = v_k$ and *consistent = true*

**for** *j* **from** *1* **to** *i–1* **do** /* for each past variable */

**if** $\neg C_{ij}(x_i, x_j)$ **then**

$conf\text{-}set_i = conf\text{-}set_i \cup \{j\}$

Remove $v_k$ from $CD_i$ and set *consistent = false*

Unassign $x_i$ and **break** inner loop

**endif**

**endfor**

**if** *consistent* **then** **return** (*i+1*, *true*)

**endfor**

**return** (*i*, *false*)

**end** *cbj-label*

# Conflict-directed backjumping: *unlabel*

## Modification to *bj-unlabel*

**function** *cbj-unlabel*(*i*)

$h$ = *max-list*(*conf-set*$_i$)
*conf-set*$_h$ = (*conf-set*$_h$ $\cup$ *conf-set*$_i$) \ {$h$}

/* Deepest variable in conflict */

**for** $j$ **from** $h+1$ **to** $i$ **do**

*conf-set*$_j$ = {$0$}

$CD_j = D_j$

**endfor**

Remove current value assigned to $x_h$ from $CD_h$

Unassign $x_h$

**if** $CD_h$ is empty **then** **return** ($h$, *false*)

**else** **return** ($h$, *true*)

**end** *cbj-unlabel*

# Backmarking

Attempts to minimize redundant consistency checks

a) $x_i$ is about to instantiated with value $v$ such that
   - $v$ had previously failed a constraint check with $x_j$
   - $x_j$'s value has not changed since that check

   $\Rightarrow$ constraint check will fail again

b) $x_i$ is about to instantiated with value $v$ such that
   - $v$ had previously succeeded in a constraint check with $x_j$
   - $x_j$'s value has not changed since that check

   $\Rightarrow$ constraint check with $j$ and its predecessors will succeed again

# Backmarking: Data structures

- $mcl_{ik}$ (initialized to *0*)
  - $n \times d$ maximum checking level array
  - $mcl_{ik}$ is the deepest variable against which $x_i = v_k$ checked
  - finer grained version of *max-check*$_i$

  $\Rightarrow$ constraint checks with variable $mcl_{ik}$ and its predecessors need to be repeated only if they have changed
- $mbl_i$ (initialized to *0*)
  - minimum backup level array
  - records the shallowest past variable that has changed since $x_i$ was the current variable

  $\Rightarrow$ variables prior to $mbl_i$ have not changed since last time

# Backmarking: *label*

## Modification of *bt-label*

**function** *bm-label*(*i*)

**for** each $v_k \in CD_i$ **do**

**if** $mcl_{ik} \geq mbl_i$ **then**

Set $x_i = v_k$ and *consistent = true*

**for** *j* **from** $mbl_i$ **to** *i–1* **do**

$mcl_{ik} = j$

**if** $\neg C_{ij}(x_i, x_j)$ **then**

Remove $v_k$ from $CD_i$ and set *consistent = false*

Unassign $x_i$ and **break** inner loop

**endif**

**endfor**

**if** *consistent* **then return** (*i+1*, *true*)

**else** Remove $v_k$ from $CD_i$

**endfor**

**return** (*i*, *false*)

# Backmarking: *unlabel*

Modification of *bt-unlabel*

**function** *bm-unlabel*(*i*)

$h = i - 1$

$CD_i = D_i$

$mbl_i = h$

**for** $j$ **from** $h+1$ **to** $n$ **do** $mbl_j = min(mbl_j, h)$

Remove current value assigned to $x_h$ from $CD_h$

Unassign $x_h$

**if** $CD_h$ is empty **then** **return** (*h*, *false*)

**else** **return** (*h*, *true*)

**end** *bm-unlabel*

# BM-CBJ: *label*

## Modification of *bm-label*

**function** *bm-cbj-label*(*i*)

**for** each $v_k \in CD_i$ **do**

**if** $mcl_{ik} \geq mbl_i$ **then**

Set $x_i = v_k$ and *consistent = true*

**for** *j* **from** $mbl_i$ **to** *i–1* **do**

$mcl_{ik} = j$

**if** $\neg C_{ij}(x_i, x_j)$ **then**

$conf\text{-}set_i = conf\text{-}set_i \cup \{j\}$

Remove $v_k$ from $CD_i$ and set *consistent = false*

Unassign $x_i$ and **break** inner loop

**endif**

**endfor**

**if** *consistent* **then return** (*i+1, true*)

**else** Remove $v_k$ from $CD_i$ and $conf\text{-}set_i = conf\text{-}set_i \cup \{mcl_{ik}\}$

**endfor**

**return** (*i, false*)

**end***bm-cbj-label*

# BM-CBJ: *unlabel*

## Modification of *cbj-unlabel*

**function** *bm-cbj-unlabel*($i$)

$h$ = *max-list*(*conf-set*$_i$)

*conf-set*$_h$ = (*conf-set*$_h$ $\cup$ *conf-set*$_i$) \ {$h$}

$mbl_i = h$

**for** $j$ **from** $h+1$ **to** $n$ **do** $mbl_j = min(mbl_j, h)$

**for** $j$ **from** $h+1$ **to** $i$ **do**

$conf\text{-}set_j = \{0\}$

$CD_j = D_j$

**endfor**

Remove current value assigned to $x_h$ from $CD_h$

Unassign $x_h$

**if** $CD_h$ is empty **then** **return** ($h$, *false*)

**else** **return** ($h$, *true*)

**end** *bm-cbj-unlabel*

# Forward checking

When $x_i = v_k$ is attempted, the domain of $x_j$ ($i < j \leq n$) is filtered

- *reductions*$_j$ (initialized to {})
  - stack in which each element is a list of values in $D_j$ that are disallowed by a previously instantiated variable
- *past-fc*$_j$ (initialized to {})
  - stack of past variables that have checked against $x_j$
- *future-fc*$_i$ (initialized to {})
  - stack of future variables against which $x_i$ has checked

*Stack operations are in tandem*

# Forward checking: *label*

## Modification of *bt-label*

**function** *fc-label*(*i*)

  **for** each $v_k \in CD_i$ **do**

    Set $x_i = v_k$ and *consistent = true*

    **for** *j* **from** *i+1* **to** *n* **do**

      **if** ¬ *check-forward*(*i*, *j*) **then**

        Remove $v_k$ from $CD_i$ and set *consistent = false*

        *undo-reductions*(*i*)

        Unassign $x_i$ and **break** inner loop

      **endif**

    **if** *consistent* **then** **return** (*i+1*, *true*)

  **endfor**

  **return** (*i*, *false*)

**end** *fc-label*

# Checking forward

**function** *check-forward*($i$, $j$)

    *reductions* = {}

    **for** each $v_k \in CD_j$ **do**

        Set $x_j = v_k$

        **if** $\neg C_{ij}(x_i, x_j)$ **then** *reductions* = *reductions* $\cup \{v_k\}$

    **endfor**

    **if** *reductions* $\neq \{\}$ **then**

        $CD_j = CD_j \setminus$ *reductions*

        *push*($j$, *future-fc*$_i$)

        *push*(*reductions*, *reductions*$_j$)

        *push*($i$, *past-fc*$_j$)

    **endif**

    **return** $CD_j \neq \{\}$

**end** *check-forward*

# Undo forward checking reductions

**procedure** *undo-reductions*(*i*)

    **while** *forward-fc*$_i$ is not empty **do**

        *j* = *pop*(*forward-fc*$_i$)

        *reductions* = *pop*(*reductions*$_j$)

        *CD*$_j$ = *CD*$_j$ $\cup$ *reductions*

        *pop*(*past-fc*$_j$)

    **endwhile**

**end** *undo-reductions*

# Forward checking: *unlabel*

Modification of *bt-unlabel*

**function** *fc-unlabel*(*i*)

*h = i – 1*

*undo-reductions*(*h*)

*update-current-domain*(*i*)

Remove current value assigned to $x_h$ from $CD_h$

Unassign $x_h$

**if** $CD_h$ is empty **then** **return** (*h*, *false*)

**else** **return** (*h*, *true*)

**end** *fc-unlabel*

# Updating current domains

**procedure** *update-current-domain*(*i*)

$CD_i = D_i$

**for** each *reductions* $\in$ $reductions_i$ **do**

$CD_i = CD_i \setminus reductions$

**end** *update-current-domain*

# FC-CBJ: *label*

## Modification of *fc-label*

**function** *fc-cbj-label*(*i*)

  **for** each $v_k \in CD_i$ **do**

    Set $x_i = v_k$ and *consistent = true*

    **for** *j* **from** *i+1* **to** *n* **do**

      **if** ¬ *check-forward*(*i*, *j*) **then**

        Remove $v_k$ from $CD_i$ and set *consistent = false*

        *undo-reductions*(*i*)

        $conf\text{-}set_i = conf\text{-}set_i \cup past\text{-}fc_j$

        Unassign $x_i$ and **break** inner loop

      **endif**

    **if** *consistent* **then** **return** (*i+1*, *true*)

  **endfor**

  **return** (*i*, *false*)

**end** *fc-cbj-label*

# FC-CBJ: *unlabel*

## Modification of *cbj-unlabel*

**function** *fc-cbj-unlabel*(*i*)

$h = max(max\text{-}list(conf\text{-}set_i),\ max\text{-}list(past\text{-}fc_i))$

$conf\text{-}set_h = (conf\text{-}set_h \cup conf\text{-}set_i \cup past\text{-}fc_i) \setminus \{h\}$

**for** *j* **from** *i* **downto** *h+1* **do**

    $conf\text{-}set_j = \{0\}$

    *undo-reductions*(*j*)

    *update-current-domain*(*j*)

**endfor**

*undo-reductions*(*h*)

Remove current value assigned to $x_h$ from $CD_h$ and unassign $x_h$

**if** $CD_h$ is empty **then** **return** (*h*, *false*)

**else** **return** (*h*, *true*)

**end** *fc-cbj-unlabel*

# Zebra problem and results

- Zebra problem from Prosser 1993
- Tables 1 and 3 from Prosser 1993